Plan całości wykładu

- Wprowadzenie (2 wykłady)
- Warstwa aplikacji (2 wykłady)
- Warstwa transportu (2-3 wykłady)
- Warstwa sieci (2-3 wykłady)
- Warstwa łącza i sieci lokalne (3 wykłady)
- Podstawy ochrony informacji (2-3 wykłady)

Plan czasowy wykładu i ćwiczeń

start

zadania programistyczne
(łącznie 16 punktów)

kolokwium (24 punktów)

zadania programistyczne i
zaliczenie ćwiczeń

egzamin (60 punktów)

Literatura do warstwy sieci

Rozdział 4, *Computer Networking: A Top-Down Approach Featuring the Internet*, wydanie 2 lub 3, J. Kurose, K. Ross, Addison-Wesley, 2004

Rozdziały 4, 7, 8, 9, 10.16, 15, 16, *Sieci komputerowe TCP/IP*, D.E. Comer, WNT, 1997

Rozdziały 4.2, 5.2, 5.3, *Sieci komputerowe – podejście systemowe*, L. Peterson, B. Davie, Wyd. Nakom, Poznań, 2000

Warstwa sieci

Cele:

- zrozumienie zasad i problemów działania usług warstwy sieci:
  - rutingu (wyboru tras)
  - skalowalności sieci
  - jak działa ruter
  - zaawansowane tematy: IPv6, mobilność
- implementacja tych zasad w Internecie

Przegląd:

- usługi warstwy sieci
- zasady działania rutingu: wybór tras
- ruting hierarchiczny
- IP
- Protokoły rutingu w Internecie
  - wewnętrzne
  - zewnętrze
- co się dzieje w ruterze?
- IPv6
- mobilność

Mapa wykładu

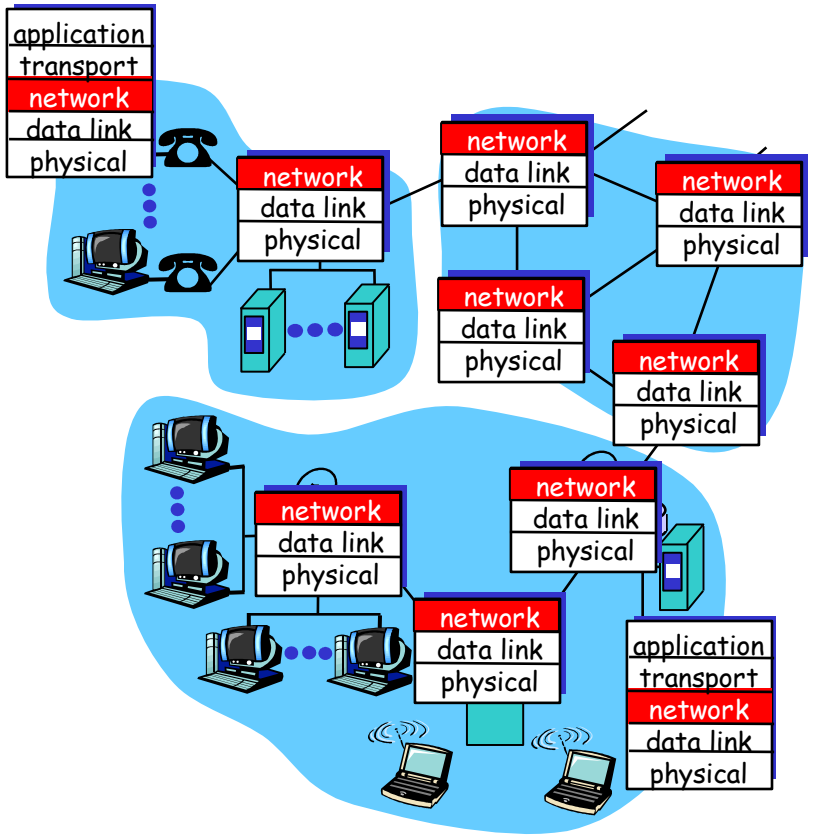
4.1 Usługi warstwy sieci z komutacją pakietów
4.2 Zasady działania rutingu
4.3 Ruting hierarchiczny
4.4 Protokół Internetu (IP)
4.5 Ruting w Internecie
4.6 Co jest w ruterze
4.7 IPv6
4.8 Ruting rozsiewczy (multicast)
4.9 Mobilność

Usługi warstwy sieci z komutacją pakietów

- skierować pakiet od hosta nadawcy do hosta odbiorcy
- protokoły warstwy sieci są w *każdym* hoście, ruterze

trzy ważne funkcje:

- *wybór ścieżki:* ścieżka, którą przebędzie pakiet od nadawcy do odbiorcy. *Algorytmy rutingu*
- *przekazywanie:* przesłanie pakietu z wejścia rutera do odpowiedniego wyjścia rutera
- *sygnalizacja:* niektóre architektury sieci wymagają komunikacji sygnalizacyjnej, zanim zostaną wysłane dane



Model usług sieciowych

Pytanie: Jaki jest *model usługi* sieci przesyłającej pakiety od nadawcy do odbiorcy?

modele usług

- best-effort?
- gwarantowana przepustowość?
- synchronizacja czasowa (zachowanie odstępów czasowych)?
- niezawodna komunikacja?
- uporządkowana komunikacja?
- informowanie nadawcy o przeciążeniu?

Najważniejsze modele usług warstwy sieci z komutacją pakietów:

wirtualne kanały albo datagramy?

Wirtualne kanały

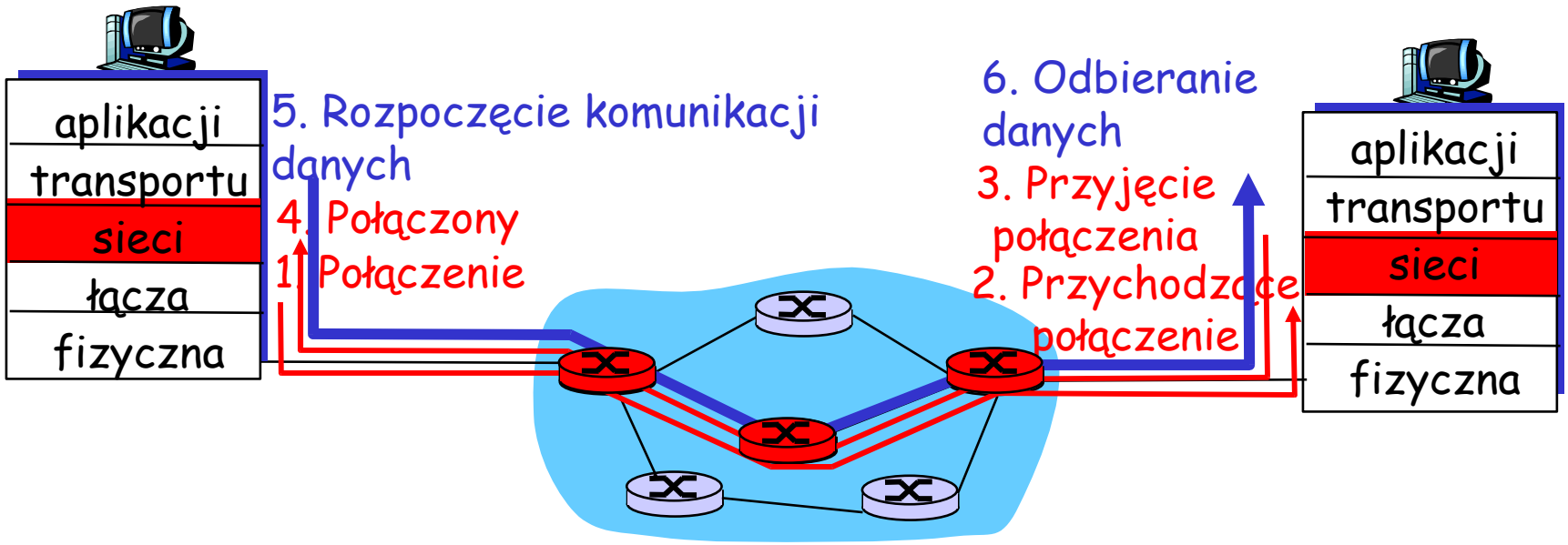
"ścieżka od nadawcy do odbiorcy zachowuje się jak kanał telefoniczny"

- z punktu widzenia wydajności
- czynności warstwy sieci dotyczą ścieżki od nadawcy do odbiorcy

- inicjalizacja wirtualnego kanału *zanim* nastąpi komunikacja danych
- każdy pakiet ma identyfikator wirtualnego kanału (nie identyfikator odbiorcy!)
- *każdy* ruter na ścieżce nadawca-odbiorca utrzymuje "stan" dla każdego wirtualnego kanału
  - połączenia w w. transportu angażowały tylko dwa hosty
- zasoby łącz i ruterów (przepustowość, bufory) mogą być *przydzielane* do wirtualnych kanałów
  - żeby uzyskać wydajność jak w kanale telefonicznym

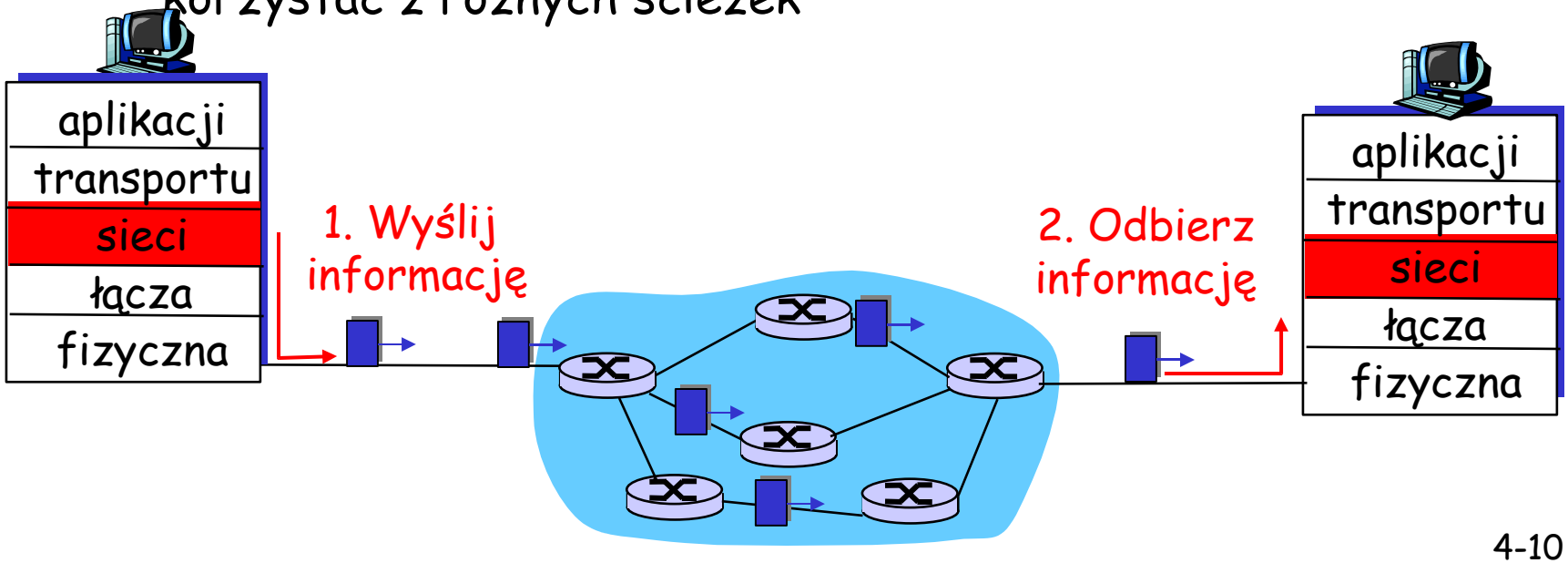
Wirtualne kanały: protokoły sygnalizacyjne

- używane do inicjalizacji, zarządzania, zamykania wirtualnego kanału
- używane w sieciach ATM, Frame-Relay, X.25
- nie używane w dzisiejszym Internecie



Sieci datagramowe: model Internetu

- nie są tworzone połączenia w warstwie sieci
- rutery: nie przechowują stanu o połączeniach koniec-koniec
  - w warstwie sieci nie jest używane pojęcie "połączenia"!
- pakiety przekazywane przy użyciu adresu odbiorcy
  - pakiety między tym samym nadawcą i odbiorcą mogą korzystać z różnych ścieżek



Modele usług warstwy sieci:

Architektura sieci	Model usług	Gwarancje ?				
		Przepusto-wość	Straty	Porząde k	Synchro-nizacja	Informacja o przeciążeni u
Internet	best effort	brak	nie	nie	nie	nie (wnioskowa-
ATM	CBR	stała	tak	tak	tak	na ze strat) nie ma
ATM	VBR	gwaranto-wana	tak	tak	tak	przeciążeni a
ATM	ABR	gwaranto-wane minimum	nie	tak	nie	nie ma przeciążeni
ATM	UBR	brak	nie	tak	nie	a tak

- Model Internetu jest rozszerzany: IntServ, DiffServ

nie

Różnice pomiędzy sieciami z wirtualnymi kanałami i sieciami datagramowymi

Internet

- komunikacja danych pomiędzy komputerami
  - usługi "elastyczne", nie ma potrzeby synchronizacji.
- "sprytne" systemy końcowe (komputery)
  - mogą się dostosować, sterować, naprawiać błędy
  - proste działanie szkieletu sieci, złożoność na "brzegu"
- wiele typów łącz
  - różne charakterystyki
  - trudno o jednolitą usługę

ATM

- wywodzi się ze telefonii
- rozmowy głosowe:
  - potrzeba synchronizacji, małego opóźnienia
  - potrzeba gwarantowanych usług
- "głupie" systemy końcowe
  - telefony
  - złożoność w działaniu "szkieletu" sieci

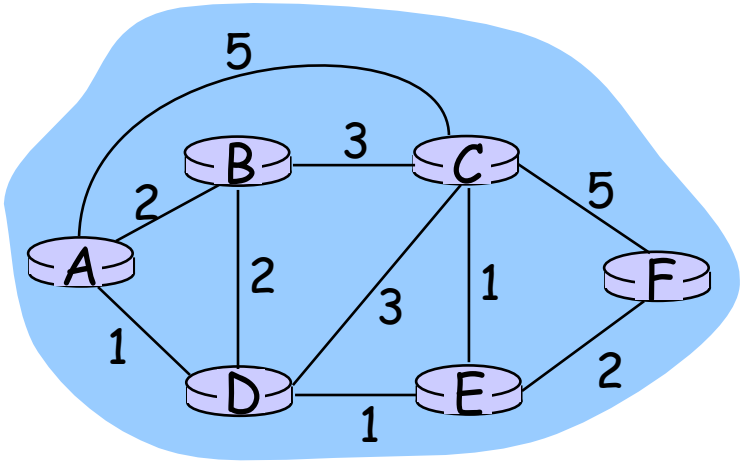
Mapa wykładu

- 4.1 Usługi warstwy sieci z komutacją pakietów
- 4.2 Zasady działania rutingu
- 4.3 Ruting hierarchiczny
- 4.4 Protokół Internetu (IP)
- 4.5 Ruting w Internecie
- 4.6 Co jest w ruterze
- 4.7 IPv6
- 4.8 Ruting rozsiewczy (multicast)
- 4.9 Mobilność

Ruting

Protokół rutingu

Cel: znajdź "dobrą" ścieżkę (ciąg ruterów) przez sieć od nadawcy do odbiorcy.



Abstrakcyjne przedstawienie rutingu na grafach:

- węzłami grafu są rutery
- krawędziami grafu są łącza sieci
  - koszt krawędzi: opóźnienie, koszt pieniężny, lub obciążenie

- "dobra" ścieżka:
  - zwykle ścieżka o najmniejszym koszcie
  - inne definicje są możliwe

Klasyfikacja algorytmów rutingu

Informacja globalna czy zdecentralizowana?

Globalna:

- wszystkie rutery mają pełną topologię sieci, koszty łącz
- algorytmy "stanu łącza"

Zdecentralizowana:

- ruter zna swoich sąsiadów, koszty łącz do sąsiadów
- iteracyjny proces obliczania, wymiana informacji z sąsiadami
- algorytmy "wektora odległości"

Statyczne czy dynamiczne?

Statyczne:

- ścieżki zmieniają się niezbyt często

Dynamiczne:

- ścieżki zmieniają się częściej
  - okresowe aktualizacji
  - po zmianie kosztu łącz

Algorytm rutingu stanu łącza (SŁ)

Algorytm Dijkstry

- topologia sieci, koszty łącz znane wszystkim węzłom
  - osiągane przez "rozgłaszanie stanu łącza"
  - wszystkie węzły mają tę samą informację
- oblicza ścieżki najmniejszego kosztu od jednego węzła ('źródła") do wszystkich pozostałych węzłów
  - zwraca tablicę rutingu dla źródła
- iteracyjny: po k iteracjach, zna najtańsze ścieżki do k celów

Notacja:

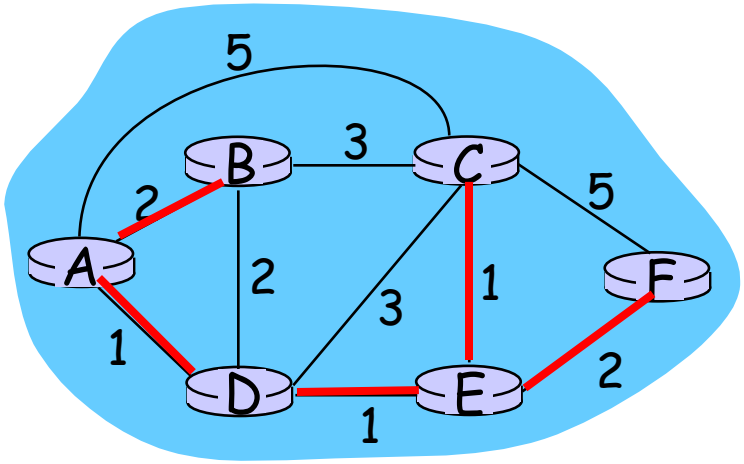
- A: źródło
- c(i,j): koszt łącza z węzła i do j. Koszt jest nieskończony, gdy węzły nie są bezpośrednimi sąsiadami
- D(v): aktualna wartość kosztu ścieżki od źródła do celu v
- p(v): węzeł poprzedzający v na ścieżce od źródła do v
- N: zbiór węzłów, do których najtańsze ścieżki są na pewno znane

Algorytm Dijsktry

1 ***Inicjalizacja:***
2 ***N*** = ***{A}***
3 dla wszystkich węzłów ***v***
4 if ***v*** sąsiaduje z ***A***
5 then ***D(v)*** = ***c(A,v)***
6 else ***D(v)*** = nieskończoność
7
8 ***Loop***
9 znajdź ***w*** nie będące w ***N*** takie, że ***D(w)*** jest najmniejsze
10 dodaj ***w*** do ***N***
11 aktualizuj ***D(v)*** dla wszystkich ***v*** sąsiednich do ***w*** i nie w ***N***:
12 D(v) = min(D(v), D(w) + c(w,v))
13 /* nowy koszt do ***v*** to albo stary koszt do ***v***, albo koszt znanej
14 najkrótszej ścieżki plus koszt od ***w*** do ***v*** */
15 ***aż wszystkie węzły będą w N***

Algorytm Dijkstry: przykład

Krok	zbiór N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
→ 0	A	2,A	5,A	1,A	nieskończ.	nieskończ.
→ 1	AD	2,A	4,D		2,D	nieskończ.
→ 2	ADE	2,A	3,E			4,E
→ 3	ADEB		3,E			4,E
→ 4	ADEBC					4,E
5	ADEBCF					



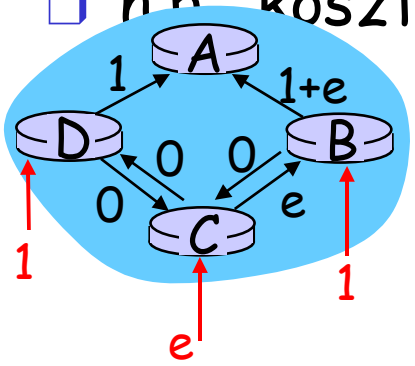
Algorytm Dijkstry: dyskusja

Złożoność algorytmu: dla n węzłów

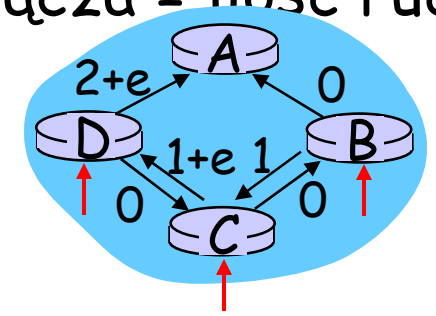
- każda iteracja: musi sprawdzić wszystkie węzły ***w***, które nie są w ***N***
- n(n+1)/2 porównań: $O(n^2)$
- bardziej wydajne implementacje możliwe: O(nlogn)

Możliwe są oscylacje:

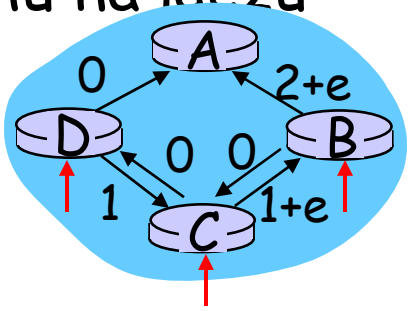
- n.p. koszt łącza = ilość ruchu na łączu



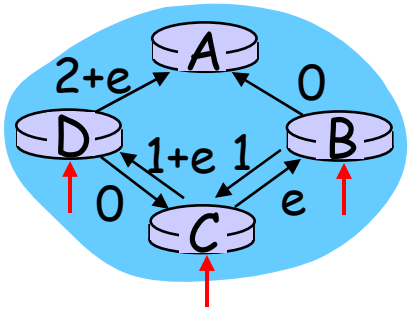
początkowo



... policz ponownie ruting



... policz ponownie



... policz ponownie

Algorytm rutingu wektora odległości (WO)

iteracyjny:

- alg. działa, dopóki węzły wymieniają informacje.
- *automatyczne zakończenie*: nie ma "sygnału" stopu

asynchroniczny:

- węzły *nie* wymieniają informacji jednocześnie!

rozproszony:

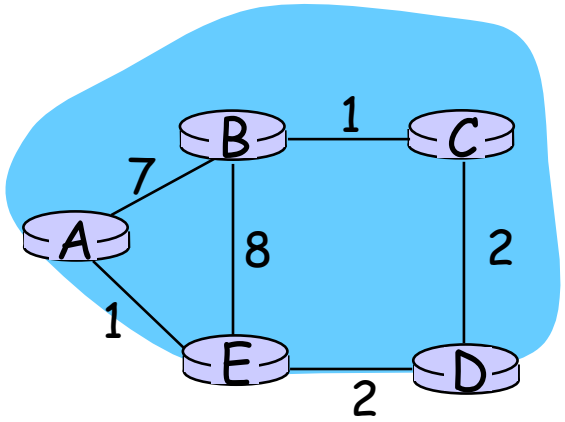
- każdy węzeł porozumiewa się *tylko* z bezpośrednimi sąsiadami

Struktura danych "wektor odległości"

- każdy węzeł ma swoją
- jeden wiersz dla każdego celu
- kolumna dla każdego bezpośredniego sąsiada węzła
- przykład: w węźle X, dla celu Y przez sąsiada Z:

$$D^X(Y,Z) = \text{odległość } od \text{ X } do \text{ Y, } przez \text{ Z jako nast. krok} = c(X,Z) + \min_w \{D^Z(Y,w)\}$$

Wektor odległości: przykład



$$D^E(C,D) = c(E,D) + \min_w \{D^D(C,w)\}$$
$$= 2+2 = 4$$

$$D^E(A,D) = c(E,D) + \min_w \{D^D(A,w)\}$$
$$= 2+3 = 5$$ pętla!

$$D^E(A,B) = c(E,B) + \min_w \{D^B(A,w)\}$$
$$= 8+6 = 14$$ pętla!

koszt do celu przez

$D^E()$	A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

cel

Tablica odległości daje tablicę rutingu

koszt do celu przez

$D^E()$		A	B	D
cel	A	1	14	5
	B	7	8	5
	C	6	9	4
	D	4	11	2

Następne łącze na ścieżce do celu, koszt łącza

cel	A	A,1
	B	D,5
	C	D,4
	D	D,4

Tablica odległości → Tablica rutingu

Ruting wektora odległości: przegląd

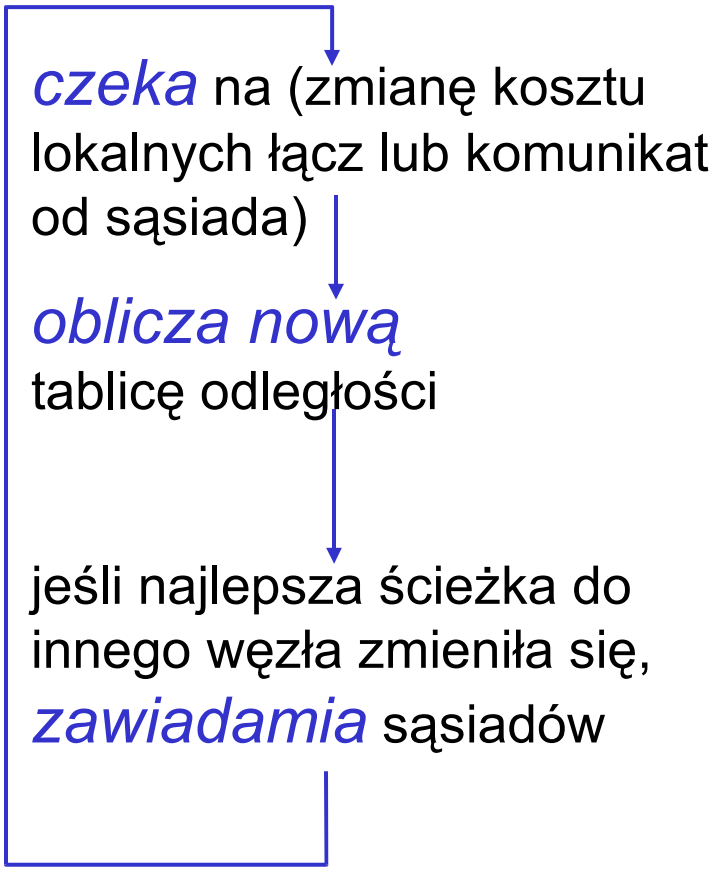
Iteracyjny, asynchroniczny:
każda lokalna iteracja powodowana przez:

- lokalną zmianę kosztów łącz
- komunikat od sąsiada: zmiana najlepszej ścieżki od sąsiada do innego węzła

Rozproszony:

- każdy węzeł zawiadamia *tylko* sąsiadów jeśli jego najlepsza ścieżka do innego węzła zmieni się
  - następnie sąsiedzi zawiadamiają swoich sąsiadów, jeśli trzeba

Każdy węzeł:



Algorytm wektora odległości:

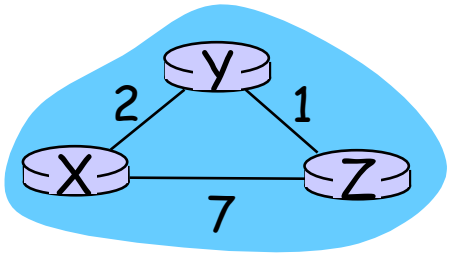
W każdym węźle X:

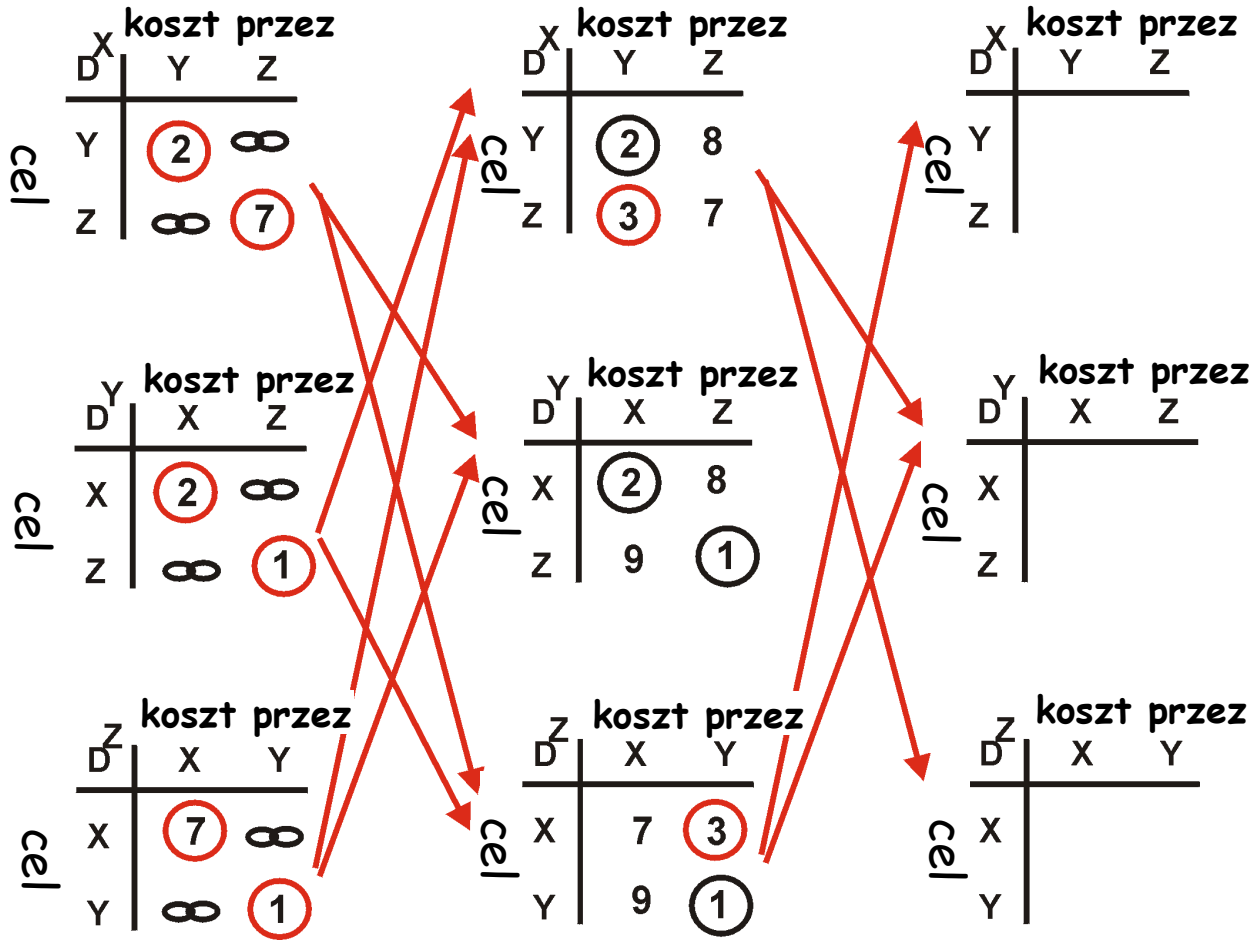
1 Inicjalizacja:
2 Dla wszystkich sąsiednich węzłów v:
3 $D^X(*,v)$ = nieskończoność // operator * oznacza "dla wszystkich wierszy"
4 $D^X(v,v) = c(X,v)$
5 dla wszystkich celów, y
6 wyślij $\min_w D^X(y,w)$ do każdego węzła // w to kolejni sąsiedzi X

Algorytm wektora odległości (c.d.):

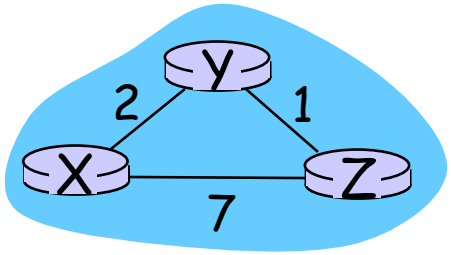
8 **loop**
9 **wait** (dopóki koszt łącza do sąsiada V nie zmieni się
10 or dopóki nie otrzymam komunikatu od sąsiada V)
11
12 **if** (c(X,V) zmieniło się o d)
13 /* zmień koszt do wszystkich celów przez v o d */
14 /* uwaga: d może być ujemne*/
15 dla wszystkich celów y: $D^X(y,V) = D^X(y,V) + d$
16
17 **else if** (komunikat od V o celu Y)
18 /* najlepsza ścieżka od V do Y zmieniła się */
19 /* V wysłał nową wartość swojego $\min_w D^v(Y,w)$ */
20 /* niech nowa wartość nazywa się "nowaWar" */
21 dla jednego celu y: $D^X(Y,V) = c(X,V) + nowaWar$
22
23 **if** jeśli mamy nowe $\min_w D^X(Y,w)$ dla dowolnego celu Y
24 wysyłam nową wartość $\min_w D^X(Y,w)$ do wszystkich sąsiadów
25
26 **forever**

Algorytm wektora odległości: przykład





Algorytm wektora odległości: przykład



X koszt przez

cel	Y	Z
Y	2	∞
Z	∞	7

X koszt przez

cel	Y	Z
Y	2	8
Z	3	7

Y koszt przez

cel	X	Z
X	2	∞
Z	∞	1

Z koszt przez

cel	X	Y
X	7	∞
Y	∞	1

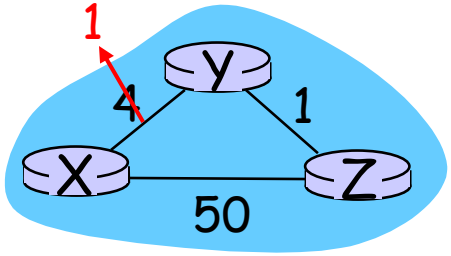
$$D^X(Y,Z) = c(X,Z) + \min_w \{D^Z(Y,w)\}$$
$$= 7+1 = 8$$

$$D^X(Z,Y) = c(X,Y) + \min_w \{D^Y(Z,w)\}$$
$$= 2+1 = 3$$

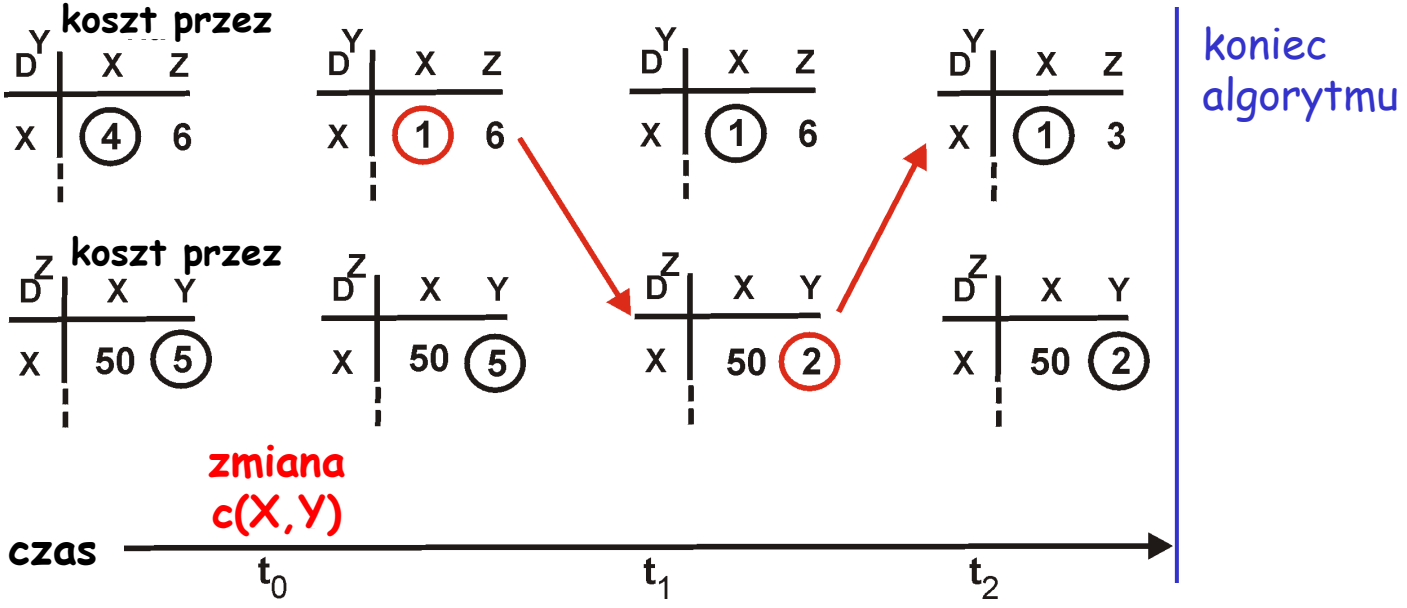
Wektor odległości: zmiany kosztów łącz

Zmiany kosztów łącz:

- węzeł wykrywa lokalną zmianę kosztów łącz
- aktualizuje tablicę odległości (linia 15)
- jeśli koszt najlepszej ścieżki się zmienił, zawiadamia sąsiadów (linie 23,24)



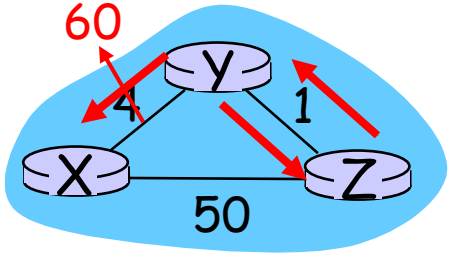
"dobre wieści szybko się rozchodzą"



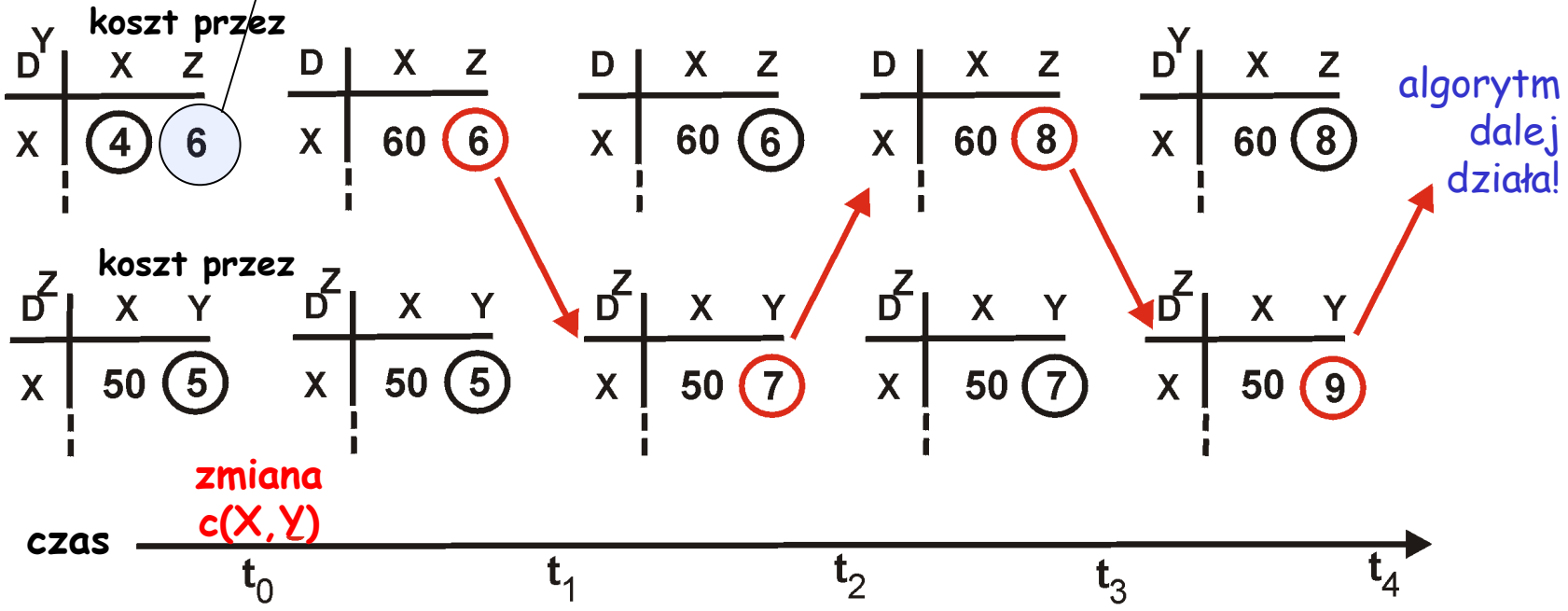
Wektor odległości: zmiany kosztów łącz

Zmiany kosztów odległości:

- dobre wieści szybko się rozchodzą...
- ...a złe wieści, powoli – problem "odliczania w nieskończoność"!



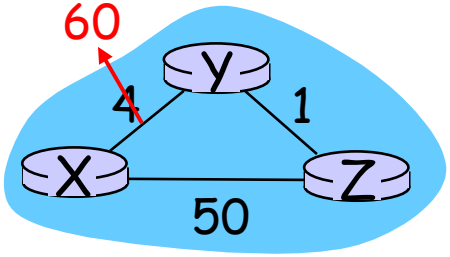
Dlaczego tu jest *6???*

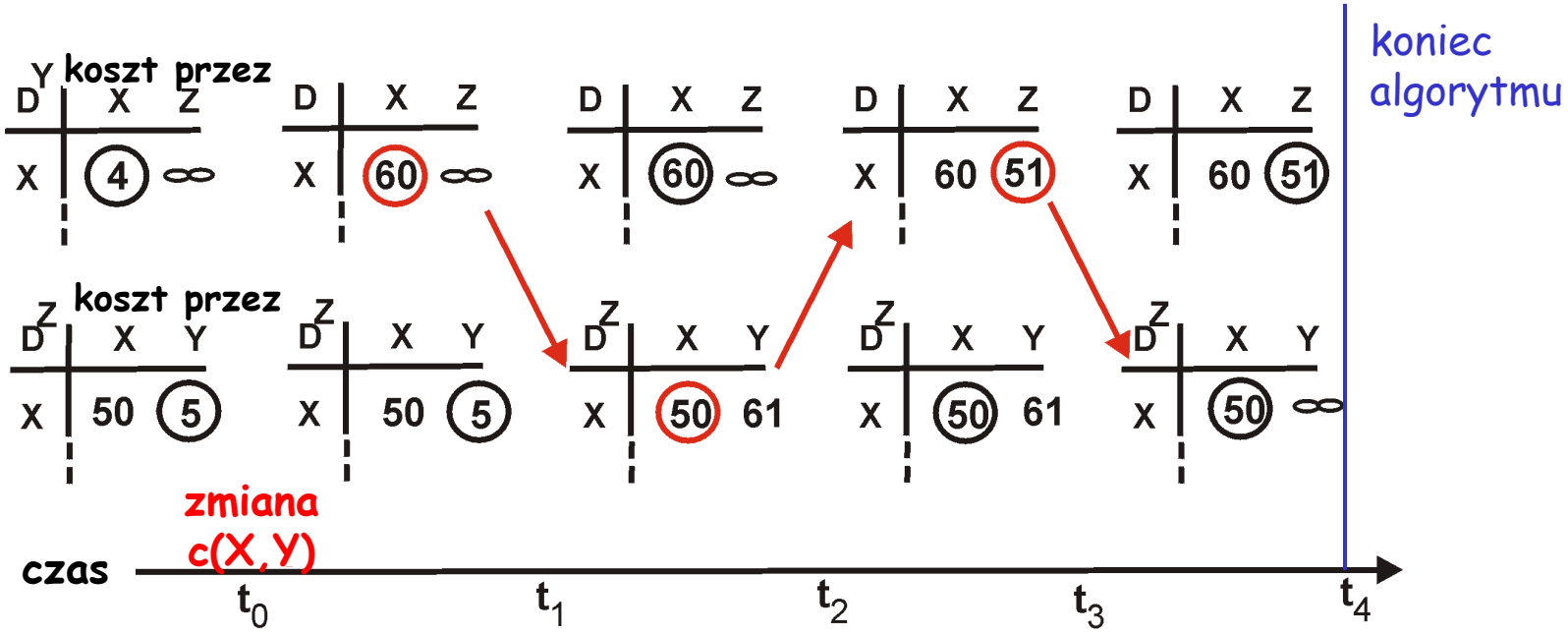


Wektor odległości: zatruty powrót

Jeśli Z rutuje przez Y do X :

- Z powie Y, że odległość (z Z) do X jest nieskończona (żeby Y nie rutował do X przez Z)
- czy to całkiem rozwiązuje problem odliczania w nieskończoność?





Porównanie algorytmów SŁ i WO

Złożoność komunikacji

- SŁ: dla n węzłów, E łączy, O(nE) komunikatów
- WO: komunikacja tylko pomiędzy sąsiadami
  - czas zbieżności jest zmienny

Szybkość zbieżności

- SŁ: $O(n^2)$
  - może mieć oscylacje
- WO: czas zbieżności zmienny
  - mogą wystąpić pętle
  - problem odliczania w nieskończoność

Odporność: co się stanie w razie awarii rutera?

LS:

- węzeł może rozgłosić nieprawdziwy koszt *łącza*
- każdy węzeł oblicza tylko *swoją* tablicę

DV:

- węzeł może rozgłosić nieprawdziwy koszt *ścieżki*
- tabela każdego węzła jest używana przez inne
  - błędy propagują się przez sieć

Mapa wykładu

- 4.1 Usługi warstwy sieci z komutacją pakietów
- 4.2 Zasady działania rutingu
- 4.3 Ruting hierarchiczny
- 4.4 Protokół Internetu (IP)
- 4.5 Ruting w Internecie
- 4.6 Co jest w ruterze
- 4.7 IPv6
- 4.8 Ruting rozsiewczy (multicast)
- 4.9 Mobilność

Ruting hierarchiczny

Dotychczas w dyskusji rutingu robiliśmy upraszczające założenia

- wszystkie rutery są identyczne
- sieć jest "płaska"

… które *nie są* prawdziwe w praktyce

skala: przy 200 milionach celów:

- nie można przechowywać wszystkich celów w tablicach rutingu!
- wymiana tablic rutingu zalała by sieć!

autonomia administracyjna

- internet = sieć sieci
- każdy administrator sieci może chcieć kontrolować ruting w swojej sieci

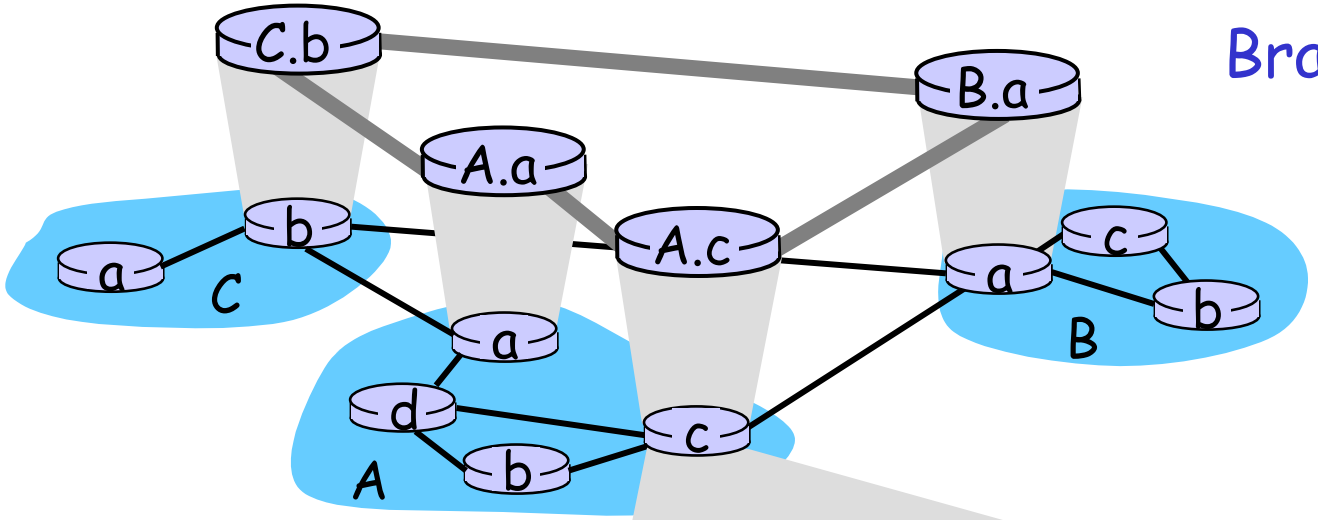
Ruting hierarchiczny

- łączy rutery w regiony, tak zwane systemy autonomiczne (ang. "autonomous systems", AS)
- rutery w tym samym AS używają tego samego protokołu rutingu
  - protokół "wewnętrzny" systemu autonomicznego
  - rutery w różnych AS mogą używać różnych protokołów rutingu wewnętrznego

rutery-bramy

- specjalne rutery w AS
- tak jak wszystkie rutery w AS, używają rutingu wewnętrznego
- są *także* odpowiedzialne za ruting do celów z poza AS
  - używają protokołu rutingu *zewnętrznego* z innymi ruterami-bramami

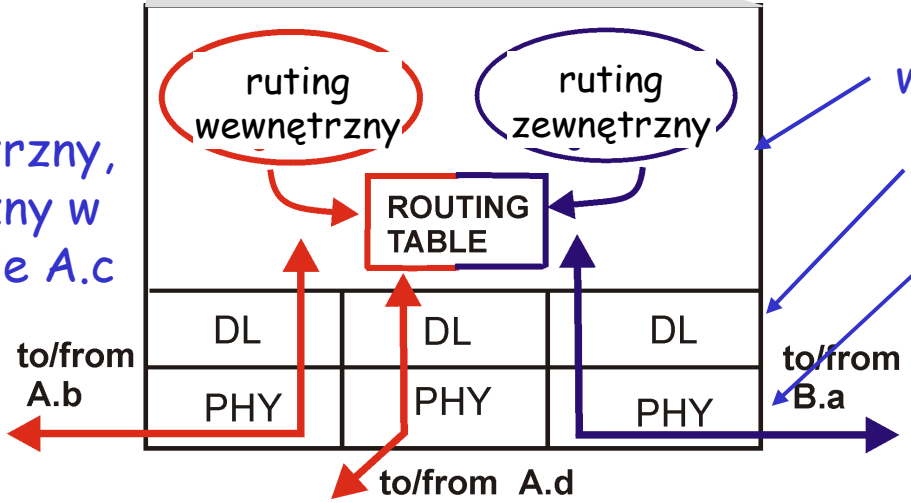
Ruting wewnętrzny i zewnętrzny



Bramy:

- pomiędzy sobą wykonują ruting zewnętrzny
- z innymi ruterami w swoim AS wykonują ruting wewnętrzny

ruting zewnętrzny, wewnętrzny w bramie A.c

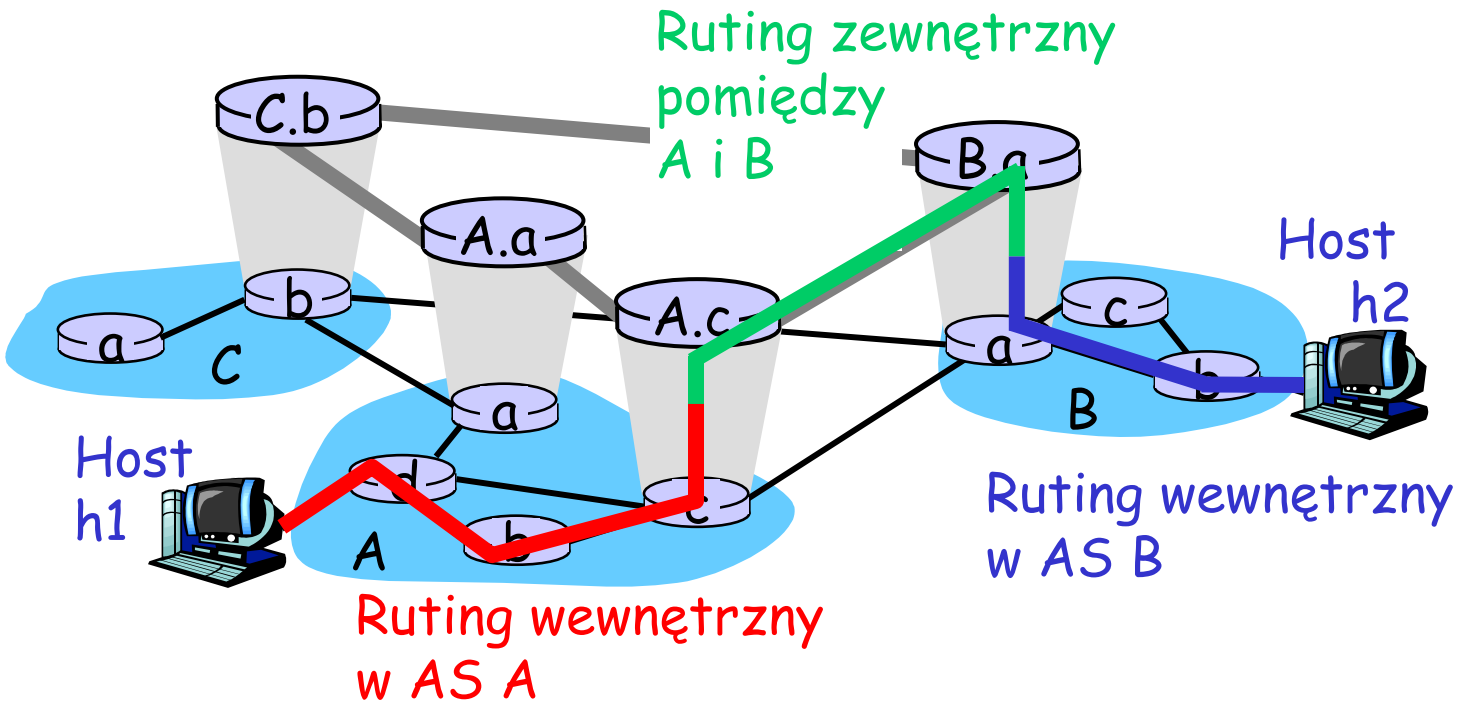


warstwa sieci

warstwa łącza

warstwa fizyczna

Ruting wewnętrzny i zewnętrzny



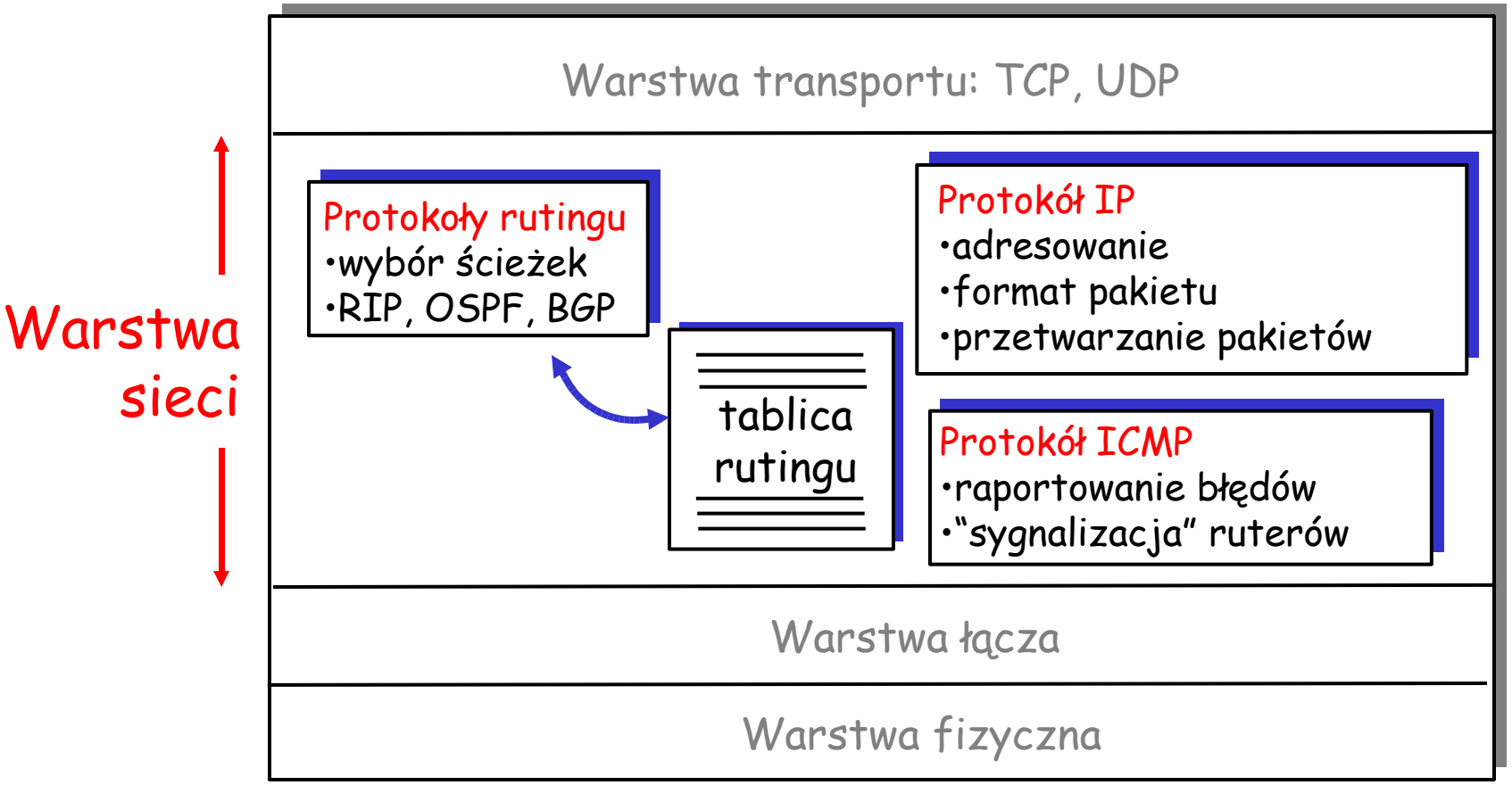
- Niedługo poznamy protokoły rutingu wewnętrznego i zewnętrznego w Internecie

Mapa wykładu

- 4.1 Usługi warstwy sieci z komutacją pakietów
- 4.2 Zasady działania rutingu
- 4.3 Ruting hierarchiczny
- 4.4 Protokół Internetu (IP)
  - 4.4.1 Adresy IPv4
  - 4.4.2 Przekazywanie pakietu od źródła do celu
  - 4.4.3 Format pakietu
  - 4.4.4 Fragmentacja IP
  - 4.4.5 ICMP: Internet Control Message Protocol
  - 4.4.6 DHCP: Dynamic Host Configuration Protocol
  - 4.4.7 NAT: Network Address Translation
- 4.5 Ruting w Internecie
- 4.6 Co jest w ruterze
- 4.7 IPv6
- 4.8 Ruting rozsiewczy (multicast)
- 4.9 Mobilność

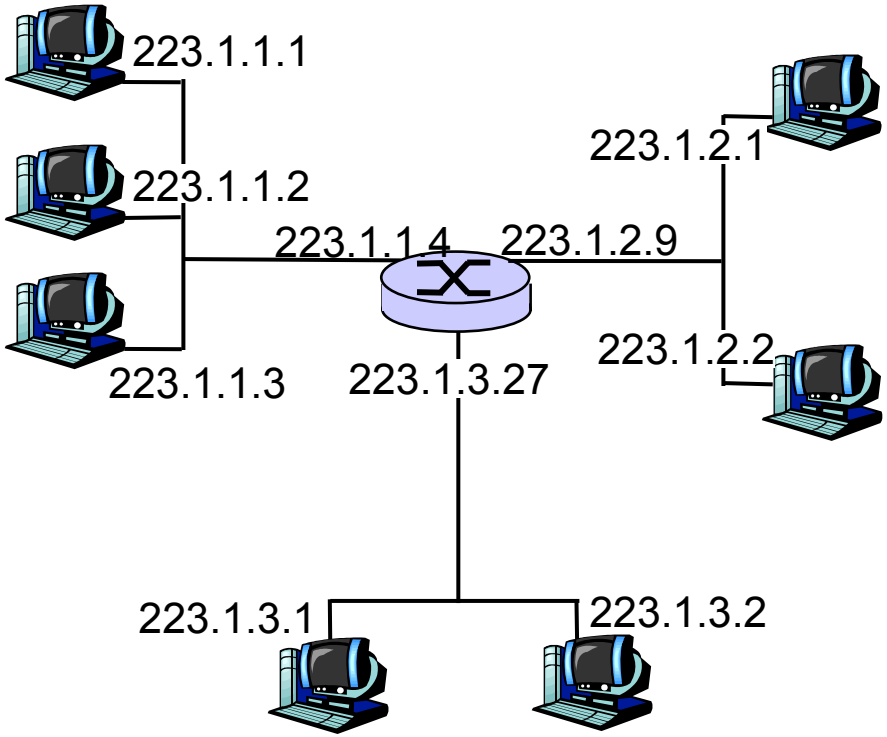
Warstwa sieci w Internecie

Funkcje warstwy sieci hosta i rutera:



Adresy IP: wstęp

- adres IP: 32-bitowy identyfikator hosta, *interfejsu* rutera
- *interfejs:* połączenie pomiędzy hostem/ruterem a łączem
  - rutery zwykle mają wiele interfejsów
  - hosty mogą mieć wiele interfejsów
  - adres IP jest związany z każdym interfejsem

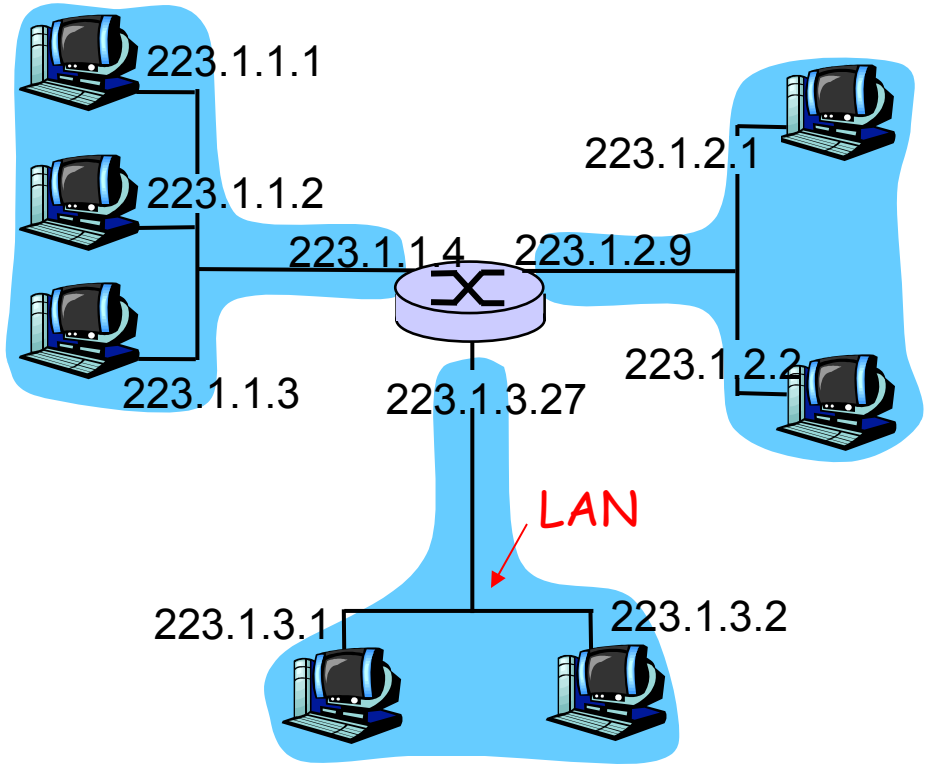


223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

Adresy IP

- adres IP:
  - część sieci (bardziej znaczące bity)
  - część hosta (mniej znaczące bity)
- *Co to jest sieć?* (z perspektywy adresów IP)
  - interfejsy urządzeń z taką samą częścią sieci adresów IP
  - które mogą się komunikować bez pośrednictwa rutera

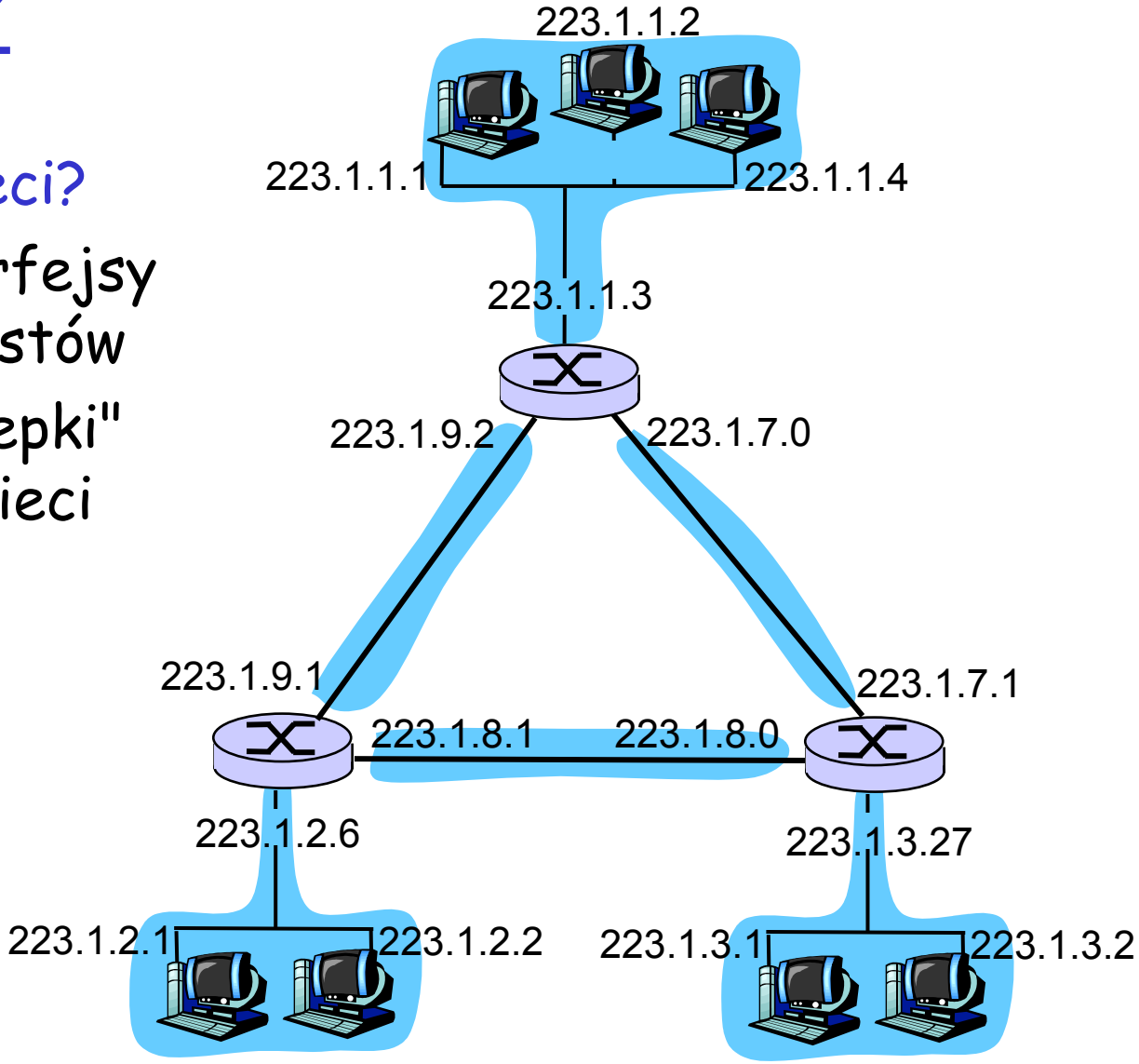


sieć składa się z 3 sieci IP (dla adresów IP zaczynających się od 223, pierwsze 24 bity są adresem sieci)

Adresy IP

Jak odróżniać sieci?

- Oddzielić interfejsy od ruterów, hostów
- stworzyć "wysepki" oddzielonych sieci

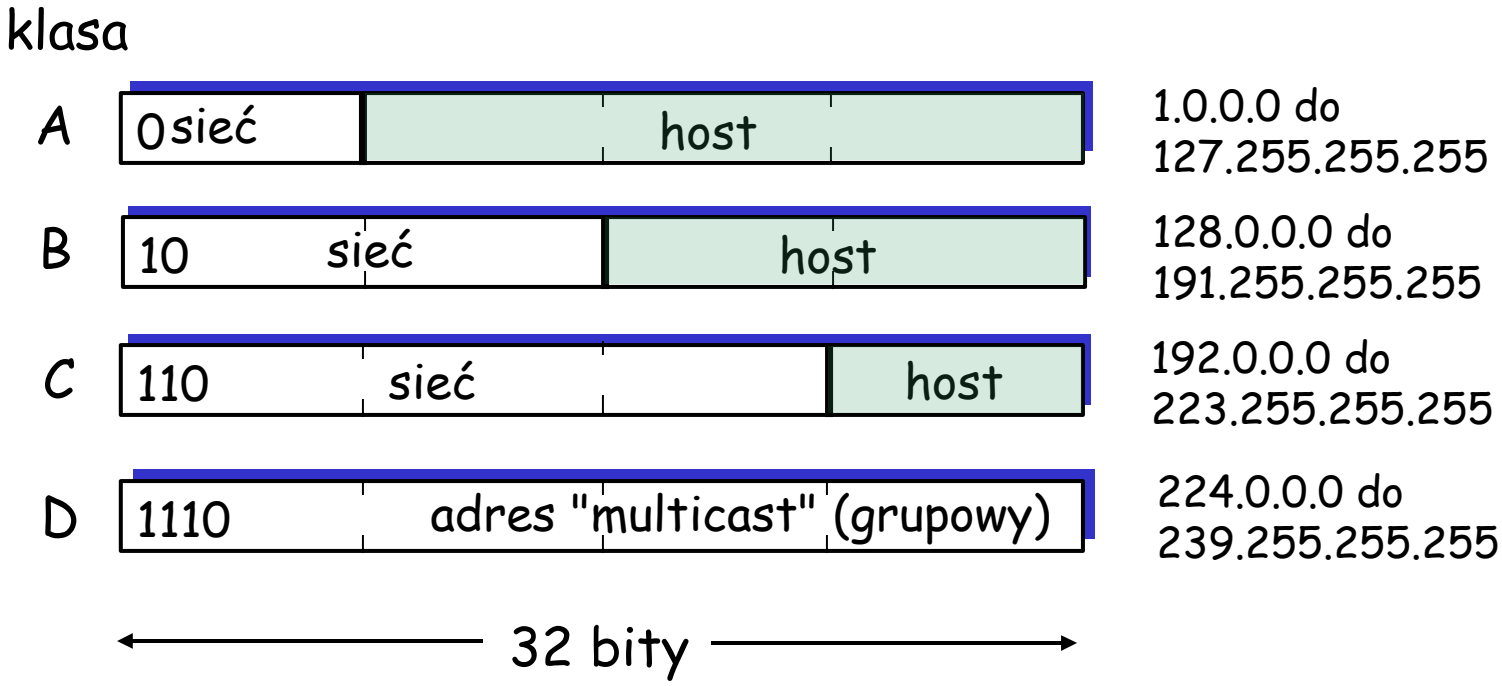


System sześciu połączonych sieci

Adresy IP

znając pojęcie "sieci", spójrzmy ponownie na adres IP:

adresowanie z "klasami":



Adresy IP: CIDR

- Adresowanie z "klasami":
  - niewydajne wykorzystanie przestrzeni adresowej, wyczerpywanie przestrzeni adresowej
  - n.p., sieć klasy B ma dość adresów dla 65K hostów, choć w sieci organizacji może się znajdować zaledwie 2K hostów
- CIDR: Classless InterDomain Routing
  - część sieciowa adresu może mieć dowolną długość
  - format adresu: a.b.c.d/x, gdzie x jest liczbą bitów w sieciowej części adresu

200.23.16.0/23

Adresy IP: jak je uzyskać?

Pytanie: Jak *host* otrzymuje adres IP?

- zapisany na stałe przez administratora w pliku
  - Wintel: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamicznie przydziela adresy IP z serwera
  - "plug-and-play"

  (więcej wkrótce)

Adresy IP: jak je uzyskać?

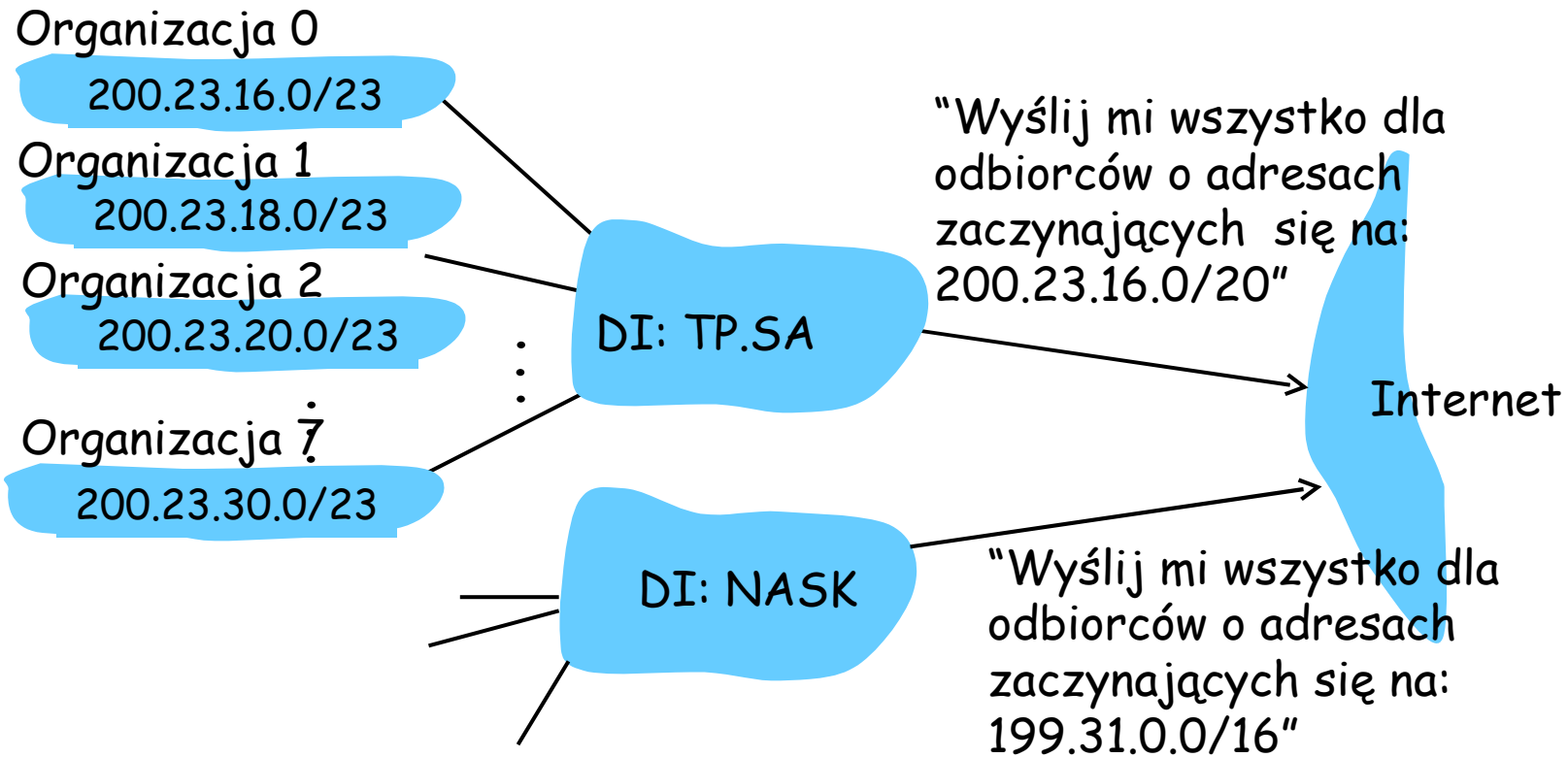
Pytanie: Jak *sieć* otrzymuje część sieciową adresu IP ?

Odpowiedź: otrzymuje przydzieloną część przestrzeni adresowej swojego DI

Adresy DI	11001000 00010111 00010000	00000000	200.23.16.0/20
Organizacja 0	11001000 00010111 0001**000**0	00000000	200.23.16.0/23
Organizacja 1	11001000 00010111 0001**001**0	00000000	200.23.18.0/23
Organizacja 2	11001000 00010111 0001**010**0	00000000	200.23.20.0/23
...	.....	....	....
Organizacja 7	11001000 00010111 0001**111**0	00000000	200.23.30.0/23

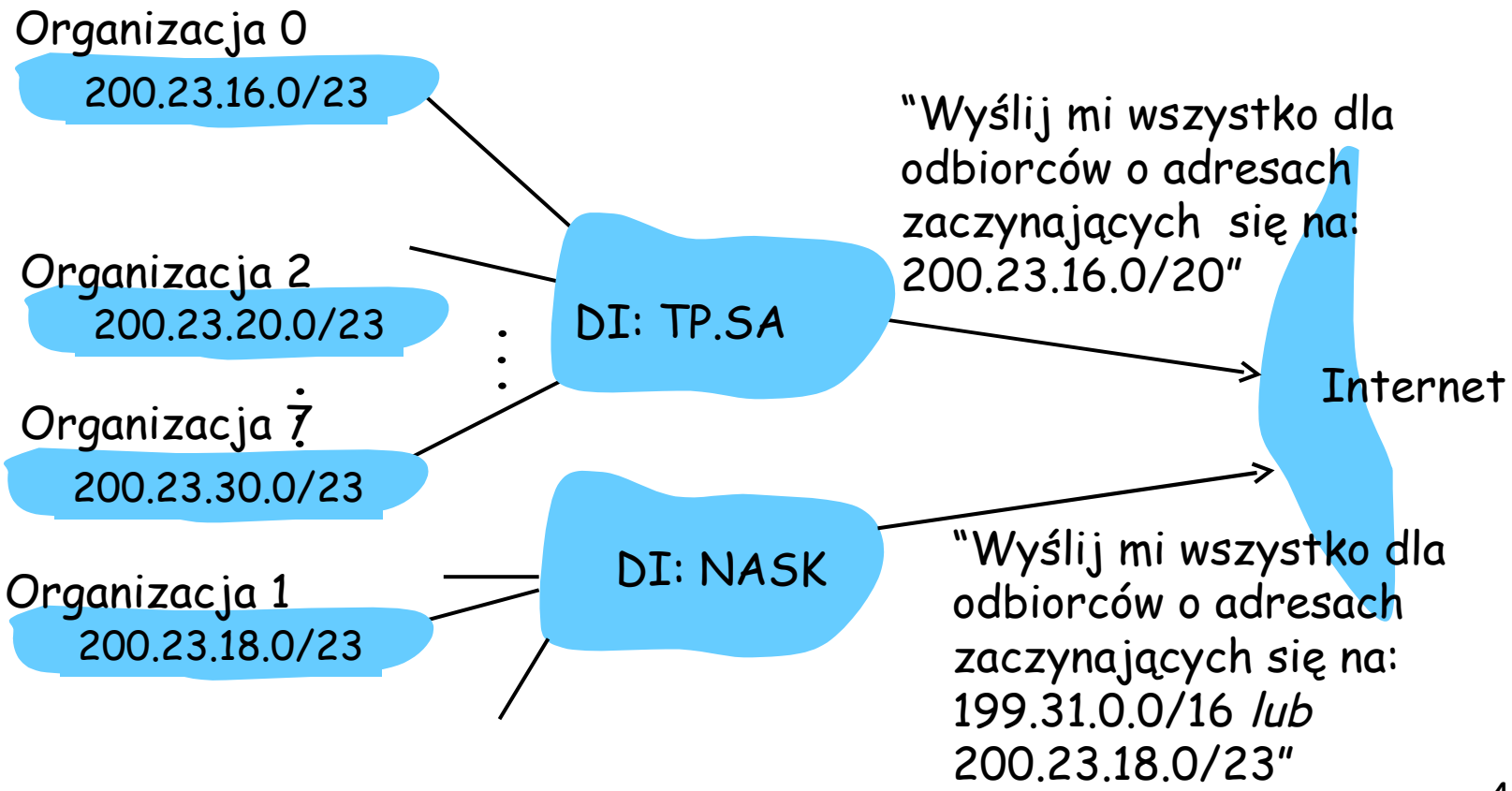
Adresowanie hierarchiczne: łączenie ścieżek

Adresowanie hierarchiczne pozwala na wydajną wymianę informacji o rutingu:



Adresowanie hierarchiczne: wybór dokładniejszych adresów

DI NASK ma ścieżkę do dokładniejszego adresu Organizacji 1



Adresy IP: ostanie słowo...

Pytanie: Jak DI otrzymuje adresy IP?

Odpowiedź: ICANN: Internet Corporation for Assigned Names and Numbers

- przydziela adresy IP
- zarządza DNS i przydziela adresy DNS
- rozstrzyga kwestie sporne